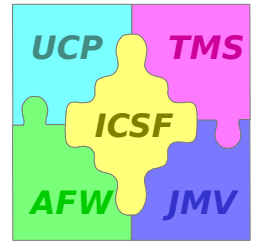


Application Framework (AFW) Symplot

Rob Peabody



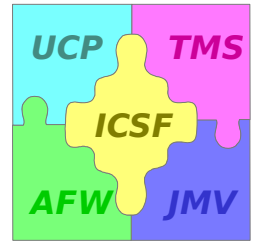
Agenda



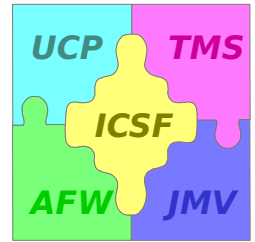
- ❑ Overview
- ❑ AFW Constructs
- ❑ Symplot Architecture
- ❑ Symplot Plug-Ins
 - Plot Controls
 - Filters
 - Views
 - Menus



AFW/Symplot Overview



- ❑ *AFW Provides support for the following features:*
 - *Menus and toolbars driven from resource files*
 - *Multiple docked toolbars in North, South, East, and West*
 - *Dockable views*
- ❑ *AFW is built on top of the Java SWING API*
 - *AFW Frame extends J Frame*
- ❑ *Symplot provides a flexible plug-in architecture for:*
 - *Controlling the display of Tracks on the ICSF map displays*
 - *Displaying tracks and associated data (labels, decorations, AOU's, etc...)*
 - *Filtering tracks from/force tracks to map displays*
 - *Plot controls (both GUI and map display)*



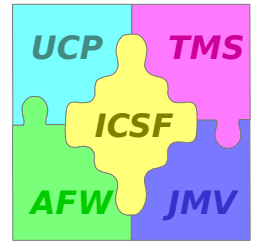
AFW Constructs

❑ *AFW Frame*

- *In addition to frameName it requires three more arguments:*
 - *className* - name of the class associated with this frame
 - *filename* - full path name of the file containing the UI properties (menu bar, toolbar, and actions in the frame)
 - *filenameDB* - full path name of the file containing the action properties

❑ *IFL Menu Builder*

- *Simplifies the process of creating menus.*
- *Builds Menus, SubMenus, and MenuItem's based on properties files*
- *Properties can be changed without recompilation*



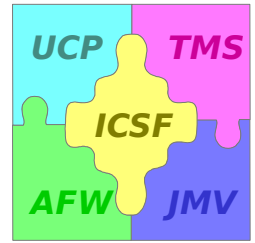
AFW Constructs (2)

❑ *UI Action Properties Files*

- *Determines the appearance and “actions” of menu bars and toolbars.*
- *Actions have many attributes, a few examples are given below*
 - *class* - The fully qualified class to launch when activated.
 - *exec* - Name of the executable to launch when activated.
 - *Feature* - Tags this action as a feature.
 - *group* - Name to provide logical grouping for actions.
 - *icon* - Name of the image file for the icon.
 - *menuActions* - Comma-separated list of the actions that go in this menu.
 - *menuLabel* - Label to use when this action appears in a menu.
 - *script* - JavaScript to be executed when this action is activated.
 - *segment* - String defining the segment this action belongs to.
 - *toolbarActions* - Comma-separated list containing the actions for this toolbar.
 - *userData* - String that is user data for this action.



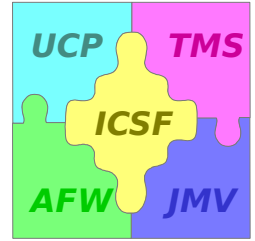
Chart Plug-in Menu Options



- ❑ *Properties files for cascading menu items and actions*
 - *ChartWindow.properties – menu items*
 - *ChartWindowDB.properties – actions behind items*
- ❑ *ChartDBInstaller is available to developers*
 - *Instantiates IFLMenuBuilder*
 - *Is used to extend the Chart Menus delivered with AFW*
- ❑ *Developer creates two text files*
 - *menuitems.properties for menu items*
 - *menuitemsDB.properties for the actions behind them*



Plug-in Menu Example



❑ **menuItems.properties**

#Test Menu

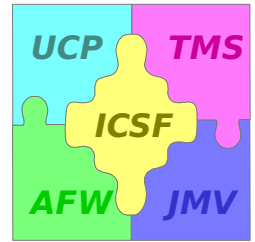
testMenu.menuActions: timeTest,dateTest,cascadeTest

cascadeTest.menuActions: levelOne

levelOne.menuActions: levelTwo



Plug-in Menu Example (2)

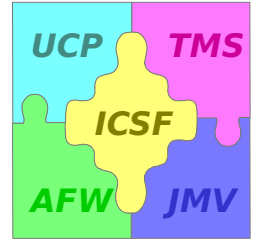


menuItemsDB.properties

```
testMenu.menuLabel: Tests
timeTest.menuLabel: Time Test
timeTest.class:      disa.afw.symplot.TimeTest
dateTest.menuLabel:  Date Test
dateTest.script:      \
    afw = Packages.disa.afw.symplot; \
    chr = application.getActiveWindow(); \
    name = javaToJavaScript(chr.getInfoBusName()); \
    dt = new afw.DateTest(name);
cascadeTest.menuLabel: Cascade Test
levelOne.menuLabel:  This is level one
levelTwo.menuLabel:  This is level two
```




Plug-in Menu Example (3)



- ❑ Use these text files in conjunction with installer:

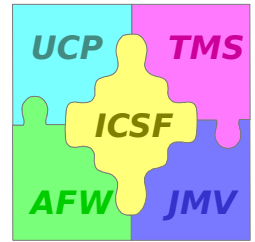
**ChartDBInstaller -DBActions
menuItemsDB.properties**

-MenuActions menuItems.properties

- ❑ *Runtime environment must be leveraged*
- ❑ *Many other items/actions can be added:*
 - *Buttons*
 - *Icons*
 - *Mnemonics*



Building Menus Sample



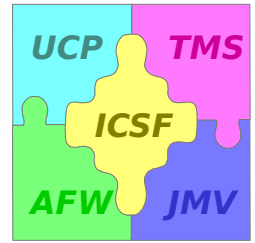
```
// BuildOn Components
JMenuBar m_menubar      = new JMenuBar();
JMenu m_menu            = new JMenu("Example Menu");
JPopupMenu m_popupMenu  = new JPopupMenu("Example Popup Menu");
JMenu m_fileMenu        = null;
JMenuItem m_zoomMenuItem = null;

// Initialize the actions manager.
IflActionManager m_manager =
    new IflActionManager(this.getClass(),"Example.properties");
IflActionManager m_managerDB =
    new IflActionManager(this.getClass(),"ExampleDB.properties");

// Build the menu bar and all its menu children.
IflMenuBuilder m_menubarBuilder =
    new IflMenuBuilder(m_menubar,null,m_manager,m_managerDB);
m_menubarBuilder.addActionListener(new myMenuBarActionListener());
m_menubarBuilder.build();
```



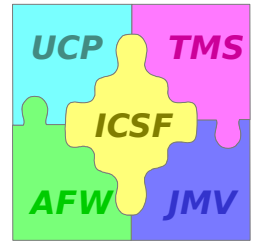
Building Menus Sample (2)



```
// Build a separate menu.  
IfIMenuBar m_menuBuilder =  
    new IfIMenuBar(m_menu,null,m_manager,m_managerDB);  
m_menuBuilder.addActionListener(new myMenuActionListener());  
m_menuBuilder.build("ExampleMenu");  
  
// Build a separate popup menu.  
IfIMenuBar m_popupMenuBuilder =  
    new IfIMenuBar(m_popupMenu,null,m_manager,m_managerDB);  
m_popupMenuBuilder.addActionListener(new myPopupActionListener());  
m_popupMenuBuilder.build("ExamplePopupMenu");
```



Building Menus Sample (3)

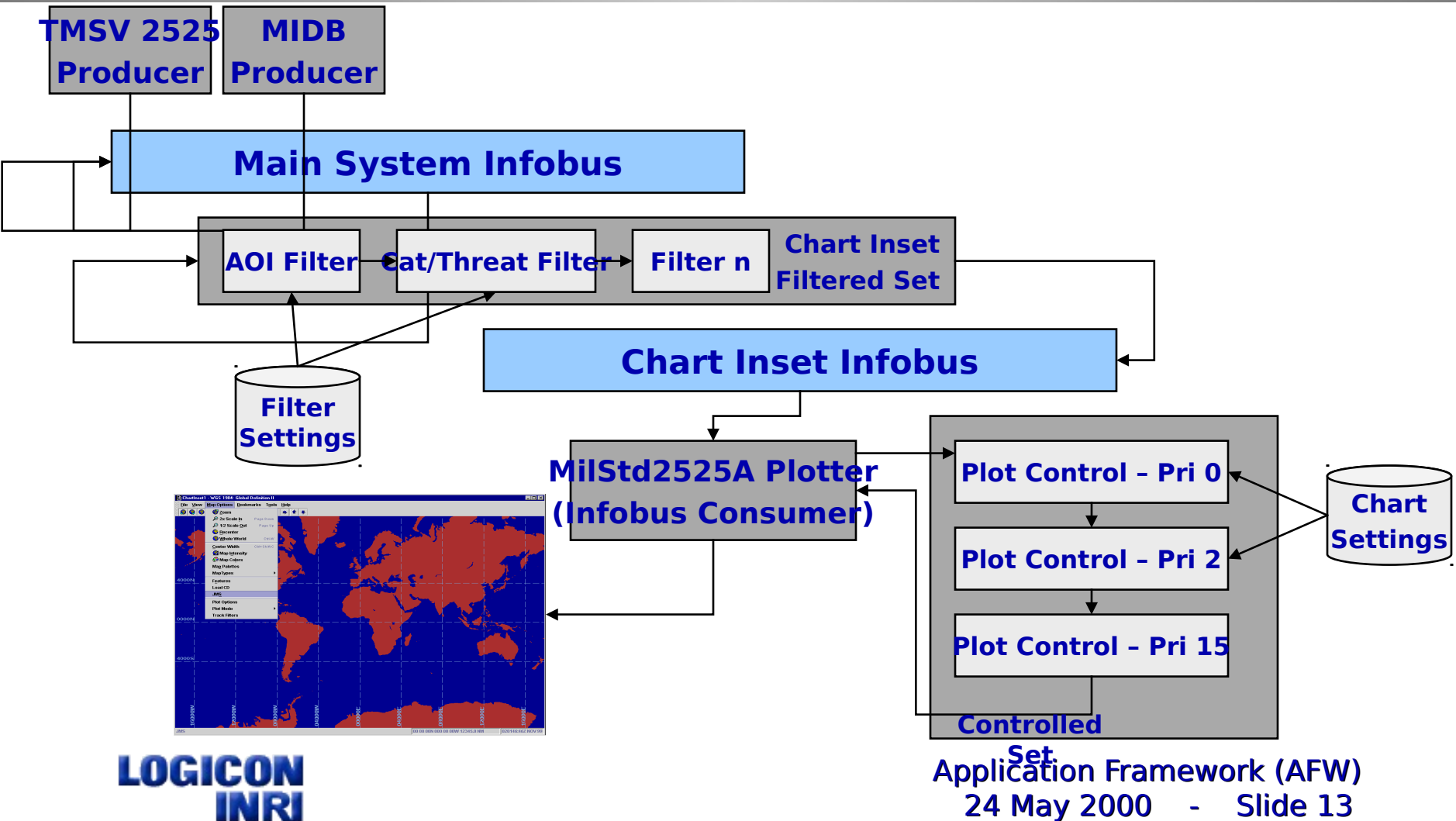
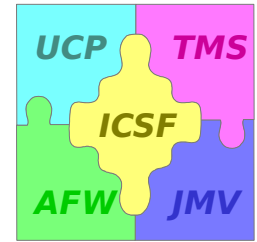


```
// Build an add-on menu and insert it onto m_popupMenu before a
// menu element name named "Close".
IflMenuBuilder m_addonMenuBuilder =
    new IflMenuBuilder(m_popupMenu,null,m_manager,m_managerDB);
m_addonMenuBuilder.addActionListener(new myAddOnActionListener());
m_addonMenuBuilder.build("AddonMenu",
                        IflMenuBuilder.ADD_BEFORE,
                        "Close");

// IflMenuService
// Get the reference of "File" JMenu
// Get the reference of "ZoomMap" JMenuItem
m_fileMenu = IflMenuService.getMenu(m_menubar,"File");
m_zoomMenuItem = IflMenuService.getMenuItem(m_menu,"ZoomMap");
try {
    m_fileMenu.setEnabled(false);
    m_zoomMenuItem.setEnabled(true);
} catch(Throwable e) {
}
```

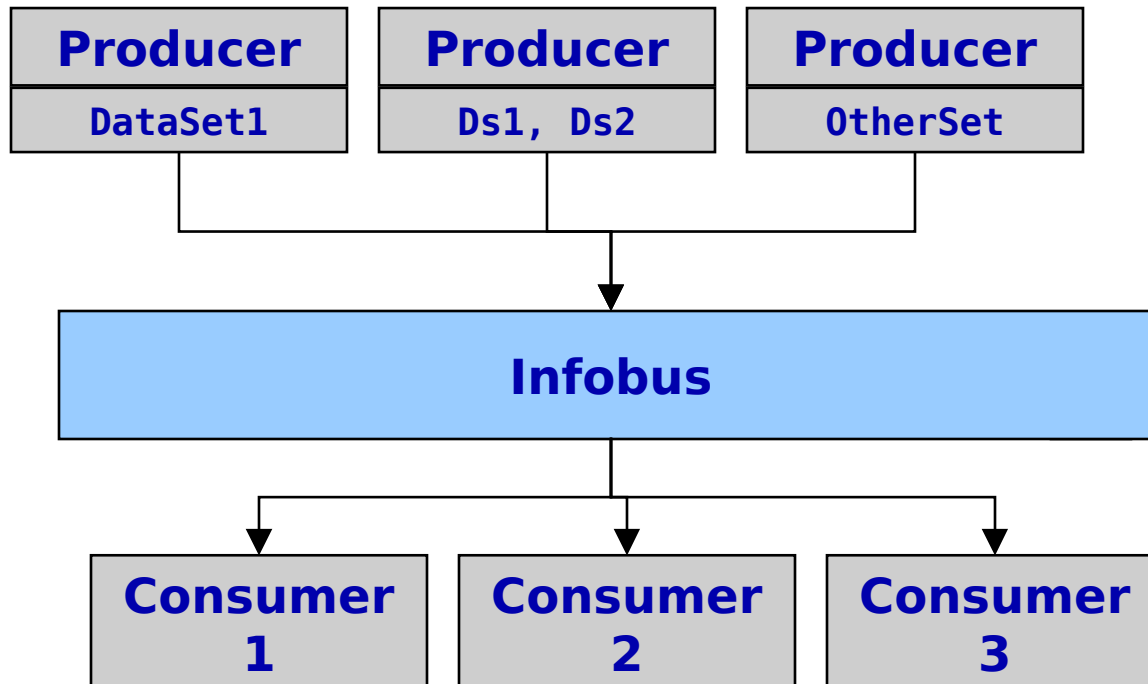
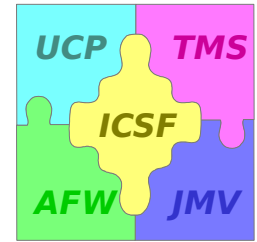


Symplot Architecture



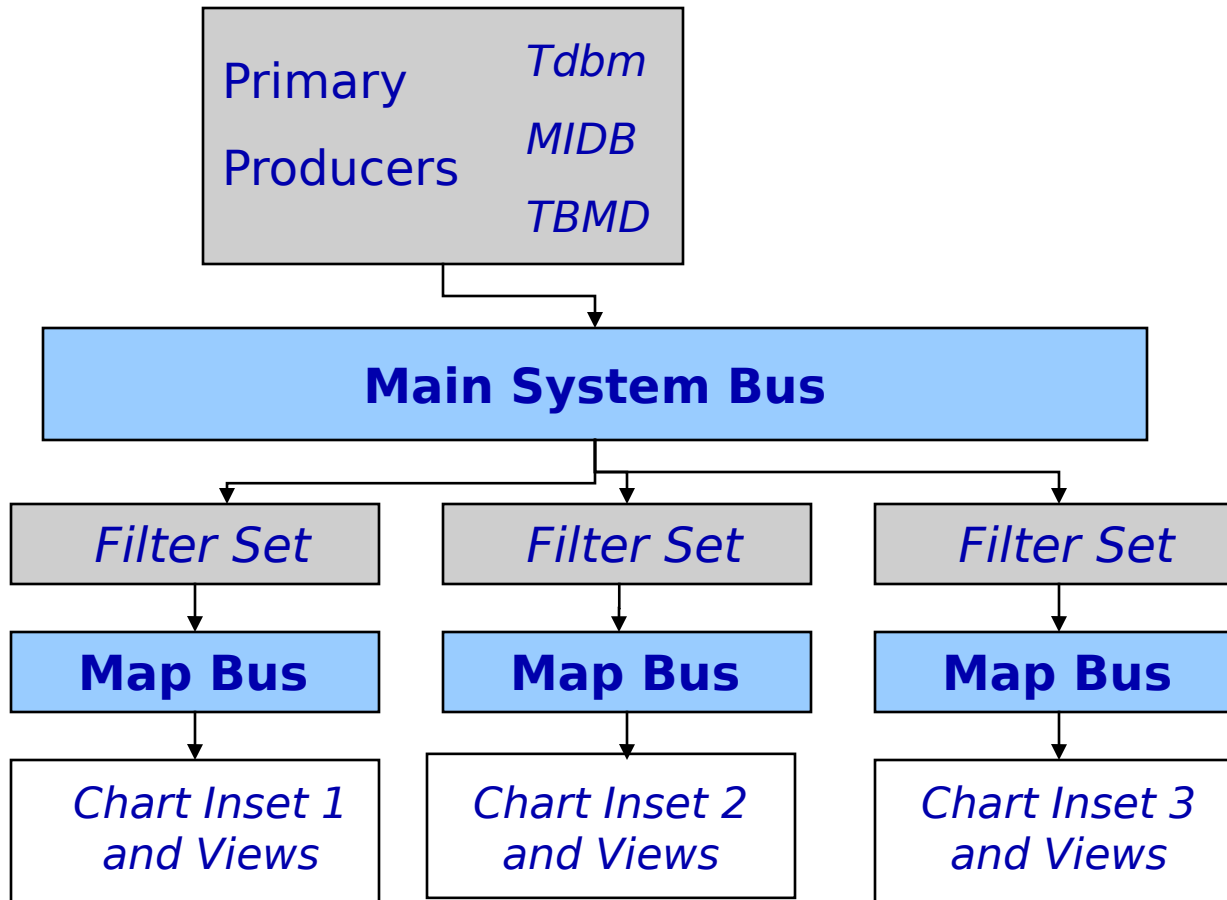
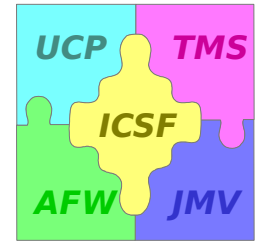


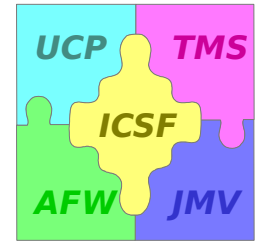
Java Infobus





Top Level Connections



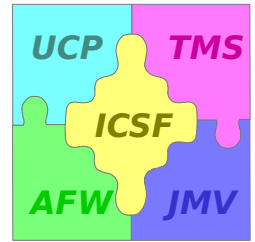


Plug-in Data Sources

- ❑ *Examples*
 - *TMS Tracks*
 - *TMS Overlays*
 - *MIDB*
 - *etc...*
- ❑ *Extend AfwAbstractInfoBusProducer*
- ❑ *Connect to Main Infobus (singletonClass in Installed.res)*
- ❑ *Tracks will be plotted with MilStd2525A Symbology if IAfwMilStd2525A is implemented*



Plug-in Data Source Producer Example

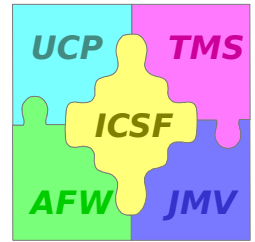


Constructor:

```
public MilStd25TrackProducer() {  
  
    super();  
    synchronized(m_AvailRevokeInterlock){  
  
        m_dataSet = new AfwSetDataItem(this, m_dataName);  
  
        for (int i=0; i<10; ++i){  
            testTrackType track = new testTrackType(i);  
            m_dataSet.add(track);  
        }  
  
    }  
}
```



Plug-in Data Source Producer Example

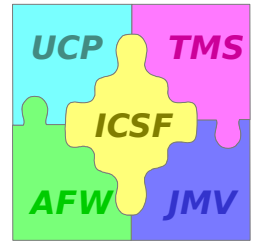


propertyChange method:
(2)

```
public void propertyChange ( PropertyChangeEvent pce ){  
  
    synchronized(m_AvailRevokeInterlock){  
  
        String s = pce.getPropertyName();  
  
        // check if the PCE refers to an InfoBus  
        if ( s.equalsIgnoreCase("InfoBus") ){  
  
            if ( pce.getSource() == this ){  
                getInfoBus().fireItemAvailable( m_dataName, flavors,  
                                                  m_producerProxy );  
            }  
            // put code here if we're watching other objects' IB properties  
        }  
        // put code here if we're watching other properties in the system  
    }  
}
```



Plug-in Data Source Producer Example



dataItemRequested method:

(3)

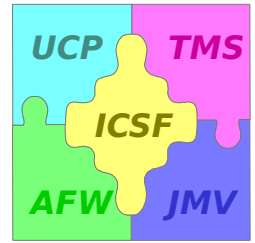
```
public void dataItemRequested ( InfoBusItemRequestedEvent ibe ){
    synchronized (m_AvailRevokeInterlock){

        if ( ibe == null ){
            return;
        }

        String s = ibe.getDataItemName();
        if (s.equals(m_dataName) {

            Trace.out.print("MilStd25TrackProducer ");
            Trace.out.println(m_dataName + " was asked for " + s);

            ibe.setDataItem(m_dataSet);
        }
    }
}
```

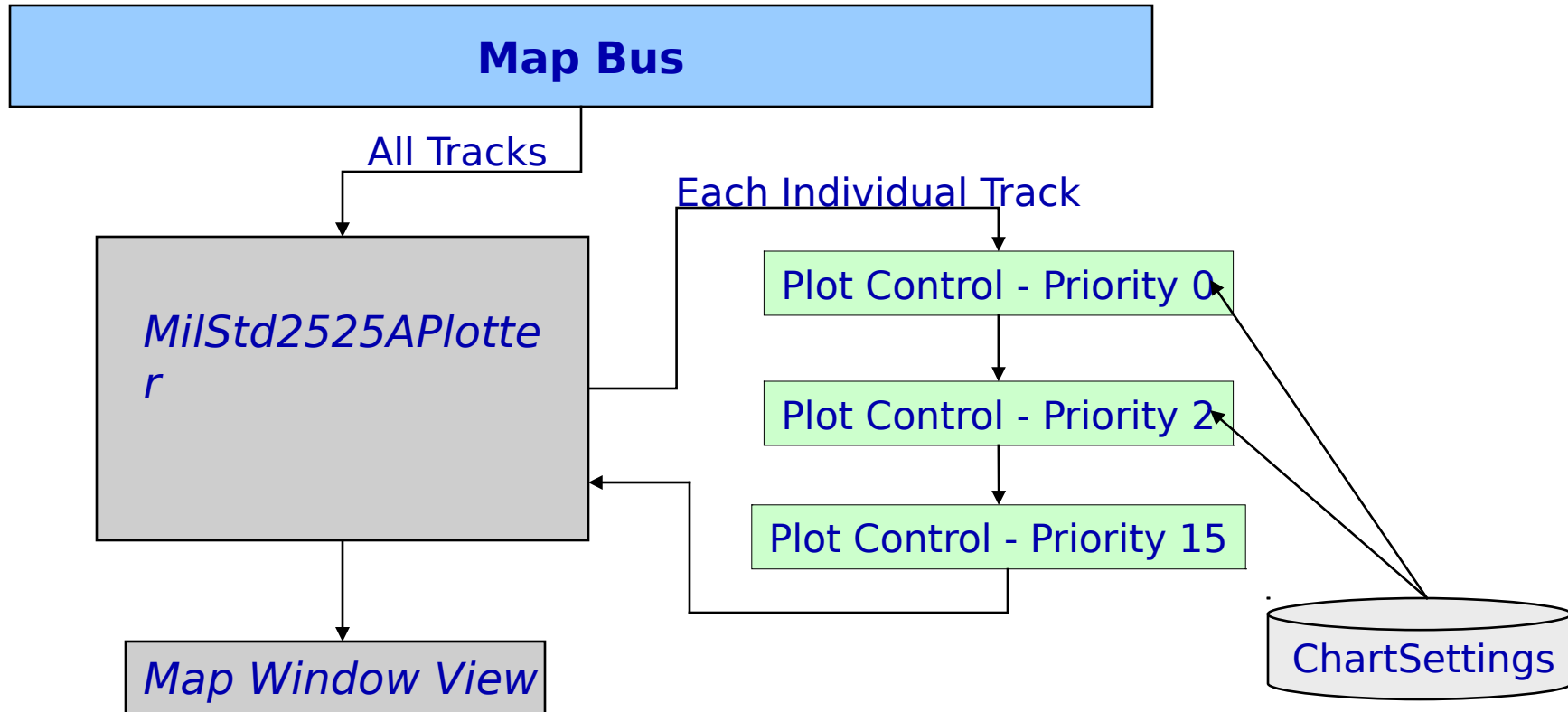
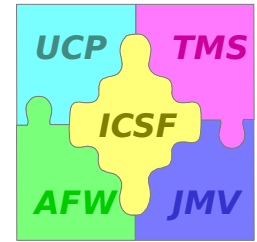


Plug-in Plot Controls

- ❑ *On, off, dots, colors, and more*
- ❑ *MIL-STD-2525A symbol control*
- ❑ *Plot controls have priority (drawing order)*
- ❑ *GUI Pages*
- ❑ *ChartSettings.properties file can be used to drive plot controls (based on GUI input)*
- ❑ *Key method for developers is `decorateSymbol()`*

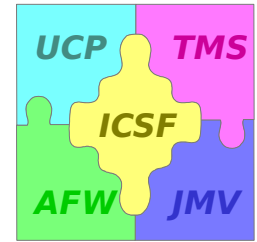


Plot Controls (details)





Plug-in Plot Control Example



Code Example:

```
public void decorateSymbol(JmvMilSymbol symbol, IAfwMilStd2525A
track) {
    MilStd2525ATestTrack tempTrack;
    if (track instanceof MilStd2525ATestTrack) {
        tempTrack = (MilStd2525ATestTrack)track;
        if (tempTrack.getCustomAttribute()) {
            JmvPenColor background = new JmvPenColor(200,
200, 200);

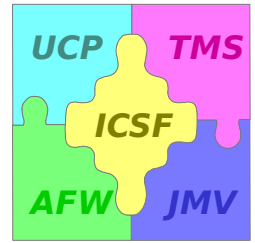
            symbol.getGraphicsModel().setBackground(background);
        }
    }
}
```

PlotControls.properties File Example:

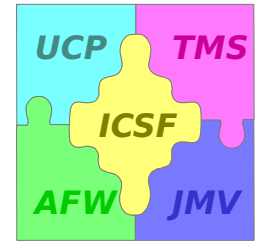
```
PlotControl.AfwMilStd2525AControl: disa.afw.symplot.AfwMilStd2525AControl
PlotControl.AfwMilStd2525AControl.priority: 0
```



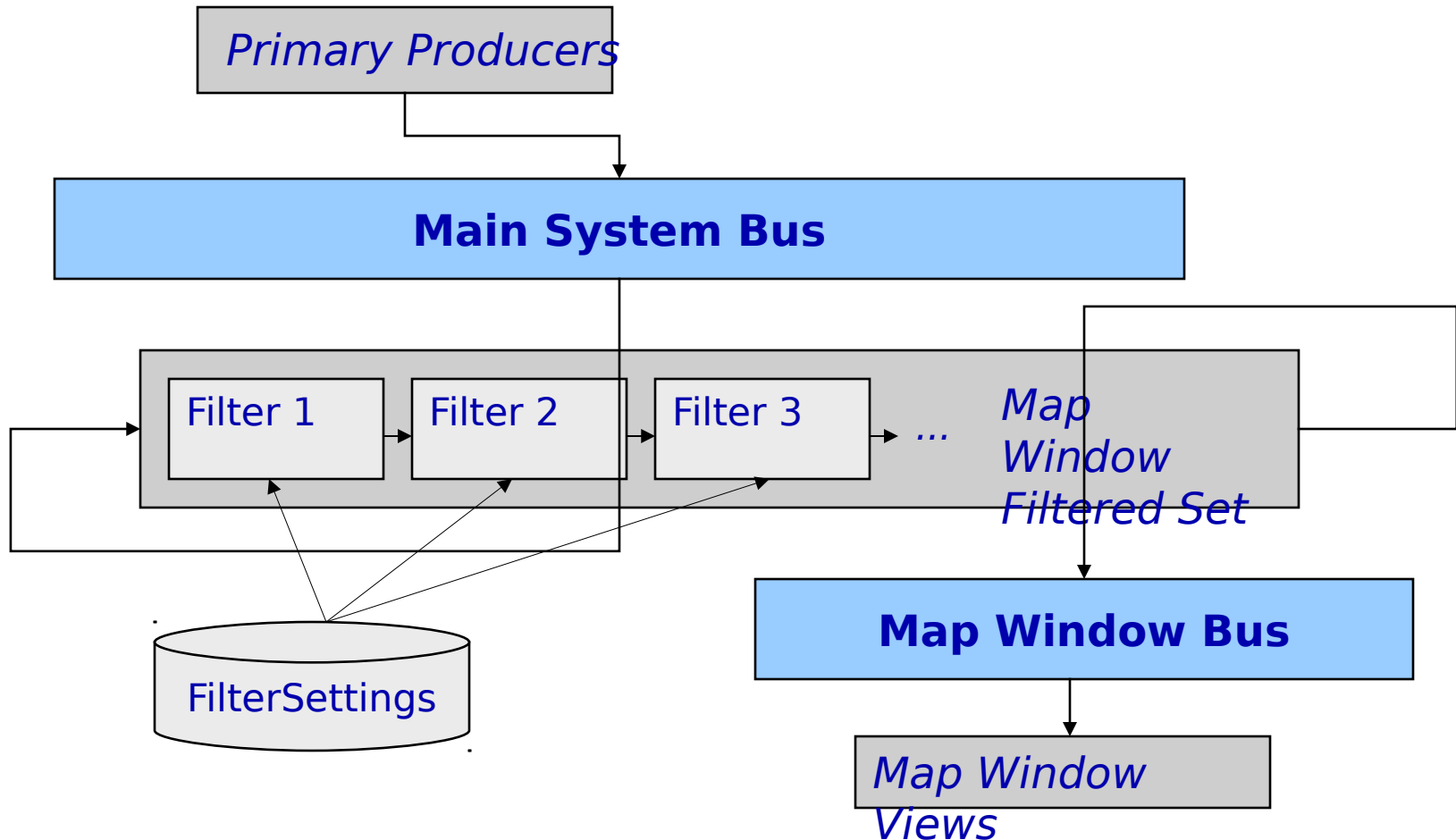
Plug-in Plot Filters



- ❑ *Remove tracks from or force plotting to map display*
- ❑ *Filter by Echelon, Affiliation, Dimension, etc...*
- ❑ *Filtered track set is maintained and visible in GUI*
- ❑ *FilterSettings.properties file can be used to drive filters (based on GUI input)*
- ❑ *Key methods for developers are pass() and force()*

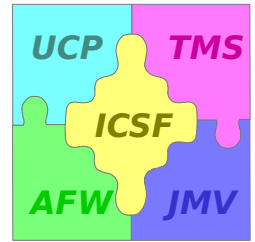


Plot Filters (details)





Plot Filter Example Code

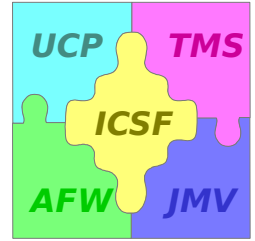


```
public boolean pass(Object o){
    if (m_filterSettings == null) {
        return true;
    }
    if (o instanceof MilStd2525ATestTrack){
        MilStd2525ATestTrack track = (MilStd2525ATestTrack)o;
        String state = m_filterSettings.getProperty(customPropertyName);
        if ("true".equals(state) && track.getCustomAttribute() == false) {
            return false;
        }
    }
    return true;
}

public boolean force(Object o) {
    boolean forced = false;
    return forced;
}
```



Plug-in View Examples

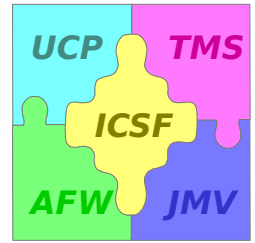


(Consumers)

- ❑ *Plotting*
- ❑ *Table*
- ❑ *Tree*
- ❑ *Property Pages*
- ❑ *Inspector*
- ❑ *Graphs*



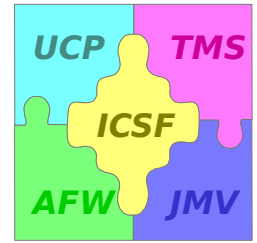
Plug-in Symbology



- ❑ *MIL-STD-2525A*
- ❑ *Naval Tactical Data System (NTDS)*
- ❑ *Theater Ballistic Missiles (TBMD Segment Support)*
- ❑ *Areas of Uncertainty (AOU)*
- ❑ *Vessel Tracking System*
- ❑ *Vector Product Format*



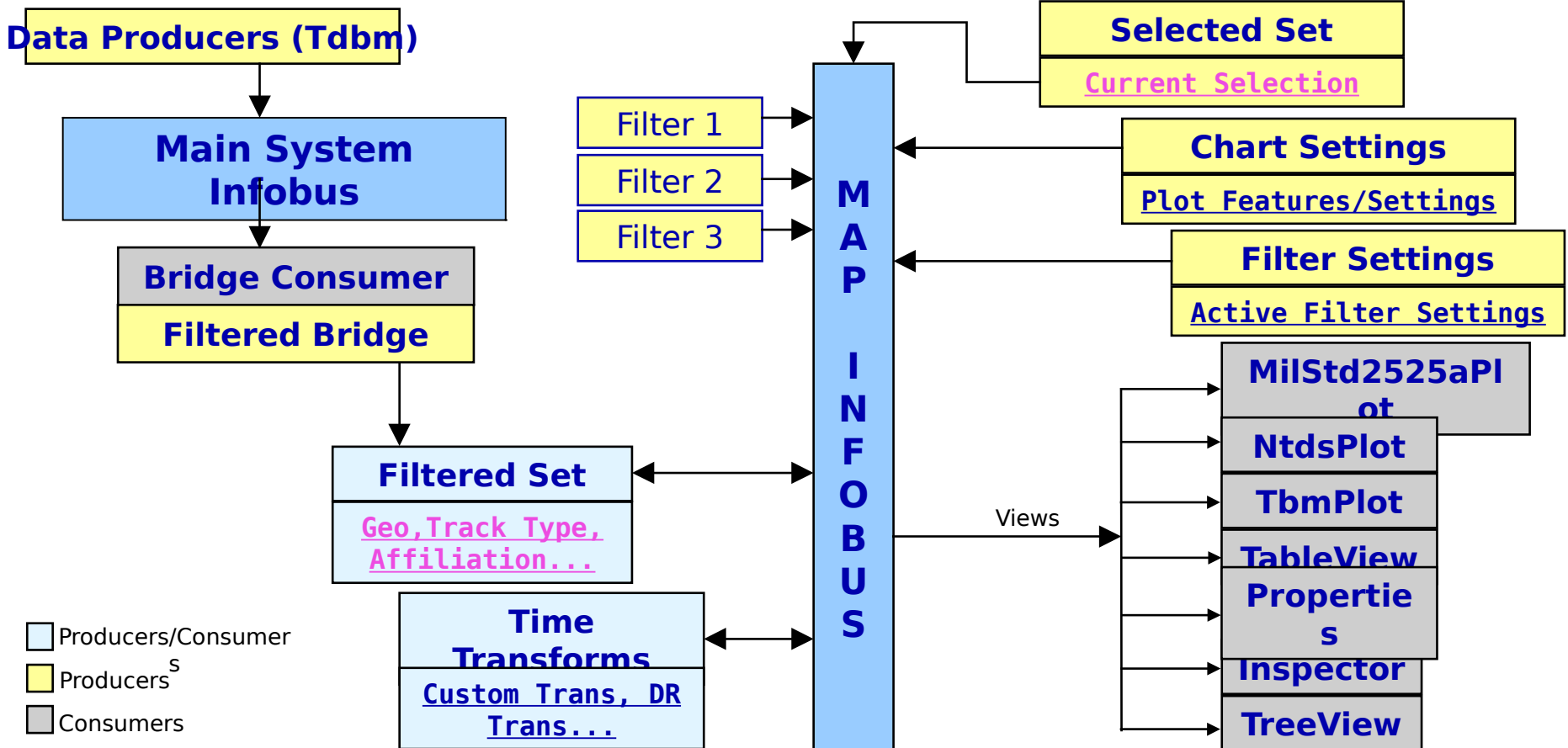
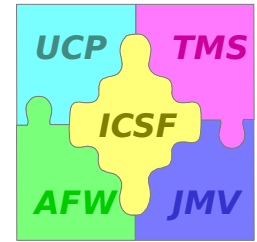
Plug-in Data Transforms



- ❑ *Dead Reckoning (DR)*
- ❑ *GEOSIT*
- ❑ *Selected Set*

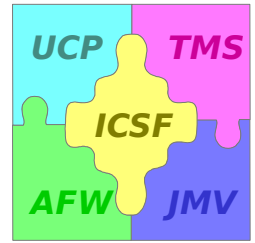


Overall System Diagram





Basic Symbolology Rendering Design



- ❑ *System setting for 2525A or NTDS*
- ❑ *MIL-STD-2525A*
 - *MilStd2525a Plotter uses Java JmvMilSymbol*
- ❑ *NTDS*
 - *NTDS Plotter uses Java JmvSymbolEx*